SKA Telescope Manager (TM): Status and Architecture Overview

Swaminathan Natarajan*[a], Domingos Barbosa [b], Joao Paulo Barraca[bc], Alan Bridger [d], Subhrojyoti Roy Choudhuri [a], Matteo DiCarlo[e], Mauro Dolci[e], Yashwant Gupta [f], Juan Guzman [g], Lize Van den Heever[h], Gerhard LeRoux [h], Mark Nicol[d], Mangesh Patil[i], Riccardo Smareglia[e], Paul Swart [h], Roger Thompson[j], Sonja Vrcic [k], Stewart Williams[d]
[a]TCS Research, Tata Consultancy Services, 54B Hadapsar I.E. Pune-411013,
[b]Instituto de Telecomunicacoes, Aveiro, Portugal,
[c]Universidade de Aviero, Aviero, Portugal,
[d]UK Astronomy Technology Centre,
[e]Instituto Nazionale di Astrofisica, Italy,
[f]National Center for Radio Astrophysics, TIFR, India,
[g]Commonwealth Scientific and Industrial Research Organization, Australia,
[h]SKA SA, National Research Foundation, South Africa
[i]Tata Consultancy Services, Pune, India,
[j]Scisys UK Ltd, Bristol, UK,
[k]HRC Herzberg Astronomy and Astrophysics, DRAO, Canada.

ABSTRACT

The SKA radio telescope project is building two telescopes, SKA-Low in Australia and SKA-Mid in South Africa respectively. The Telescope Manager is responsible for the observations lifecycle and for monitoring and control of each instrument, and is being developed by an international consortium. The project is currently in the design phase, with the Preliminary Design Review having been successfully completed, along with rebaselining to match project scope to available budget. This report presents the status of the Telescope Manager work, key architectural challenges and our approach to addressing them.

Keywords: SKA TM status, Telescope Manager, TM architecture, observation management, scalability challenges, specifications-driven approach, resource capabilities, telescope monitoring and control.

1. INTRODUCTION

The SKA radio telescope project is building two telescopes, SKA-Low and SKA-Mid, in Australia and South Africa respectively. The major characteristics of these telescopes can be perceived from the following table:

	SKA_LOW (Australia)	**SKA_MID (South Africa)**
Sensors type	512 stations of 256 dipoles each	197 Dishes (including 64 MeerKAT)
Frequency range	50-350 MHz	0.45-15 GHz
Collecting Area	0.4 Km^2	32 000m^2
Max baseline	65 Km (between stations)	150 Km
Raw Data Output	157 Tb /sec (0.49 Zetabyte/year)	3.9 Tb/sec (122 Exabyte/year)

Science Archive	0.4 PB/day (128 PB/year)	3 PB /day (1.1 Exabyte/year)

The Telescope Manager (TM) is the subsystem with responsibilities for managing the observations lifecycle, and for the monitoring and control of the telescope. This includes

- Providing facilities for scientists to prepare and submit observing proposals, and for proposal review and selection.
- Preparing observing scripts.
- Planning and scheduling telescope usage, including re-scheduling when necessary.
- Executing observations, including orchestrating the telescope subsystems and handling faults and abnormal situations, and domain specific control during observations.
- Responding to VO Events and targets of opportunity.
- Monitoring and control to manage the system health and state
- Providing user interfaces to support operations and engineering activities, including tracking the system state and managing its life cycle.
- Maintaining an archive of monitoring and control data to support diagnostics.

This paper briefly summarizes the history and status of the TM project, and provides an overview of its architecture, followed by a discussion of some of the interesting architectural aspects.

2. HISTORY AND STATUS

An initial design concept was created in 2010, as part of the SKA preparatory phase, culminating in a design concept review (CoDR). This was followed by the creation of a Work Breakdown Structure (WBS) that identified four technical areas of work: observation management, telescope control, self-management ("local monitoring and control") and infrastructure. These were complemented by three supporting areas: management, systems engineering and prototyping. This WBS served as the basis for consortium formation.

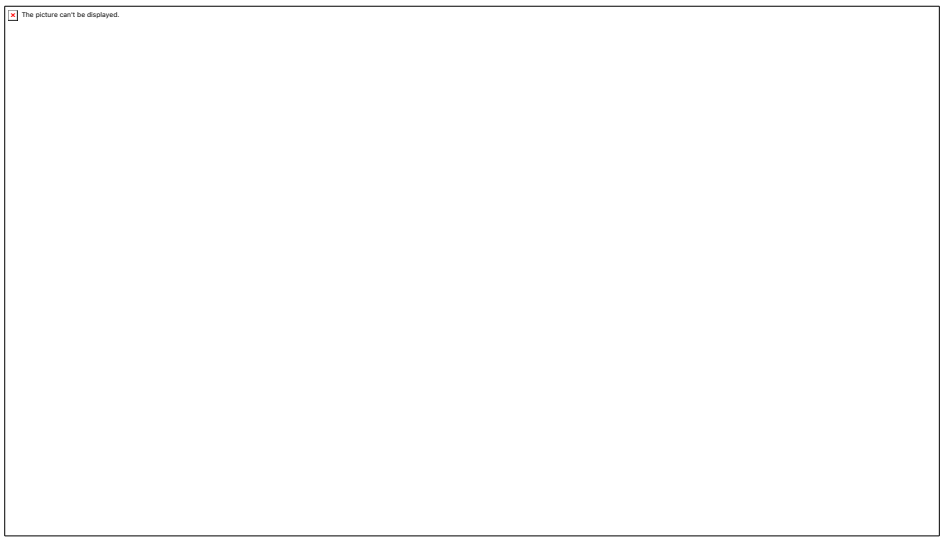

Figure 1: TM Consortium

The current project to design the TM started in Oct 2013. It was originally scheduled to take 3 years, and is currently expected to be complete by mid-2017 owing to some project delays. The design consortium is shown in Figure 1. It should be noted that in addition to the institutional responsibilities identified, individual engineers from the participating countries also contribute to some of the other work packages e.g. Italian engineers participate in observation management and user interfaces.

The first year of work focused on the development of the overall TM requirements and architecture, based on the initial versions of the system requirements for each telescope. A key challenge during this phase of work is that in addition to refinement of system requirements, the architectural concepts of all the other telescope subsystems (called "Elements") are also evolving concurrently, impacting their interfaces and demands from TM. As discussed later, this challenge was addressed by a twofold strategy: at the technical level, TM adopted a specifications-driven architecture complemented by a generic interface to other Elements, and at the system engineering level, with ICDs and iterative consistency management. This phase culminated in a Preliminary Design Review (PDR) in early 2015.

During this phase, we carefully refrained from technology choices – while options were explored, no decision was made on implementation technologies. An interesting outcome of the PDR was that the panel disagreed with this, suggesting that it would have been better to make at least the core control framework choices early in the lifecycle, since that has significant architectural impact. As discussed below, we have since made most of the technology choices, and in particular, Tango Controls[1] has been selected as the control systems framework for the project as a whole.

The project went through rebaselining in 2015, shrinking the scope to fit within available budget. This included dropping a telescope for phase 1 (a survey telescope had also been planned), reducing receiver counts, shortening baselines and reducing the number of channels. Our use of the specifications-driven approach limited the impact of these changes on TM architecture.

Recently, an operational decision was made to locate the bulk of the proposal processing, observation design and scheduling functionality at the Global Headquarters, and more importantly, to combine the proposal processing for both telescopes, to accommodate the possibility of proposals that require observations at both instruments. This had a significant impact on our design: previously we had focused on building a largely common software product that could be configured and deployed at each site. With the change, it made more sense to rework the architecture to create three Products, an observatory Product to handle proposals and the first part of the observation life cycle, and then two deployments of a telescope control Product that executes the observations and manages the instrument life cycle.

Currently, we have largely completed the (updated) requirements and architecture at the overall TM level, and developed draft requirements and architecture for each of the TM subsystems. We are currently working out the detailed design of the various components of each subsystem. We have developed prototypes of several of the components in order to validate the design and finalize technology and implementation choices. Both internal and external interfaces are being firmed up. We are also performing user experience studies to develop concepts for the various operations interfaces.

3. ARCHITECTURE OVERVIEW

The following are some of the requirements that influence the TM architecture:

- Observing modes: continuum imaging, spectral line imaging, pulsar search, pulsar timing, VLBI.
- Support partitioning of the telescope into subarrays on which observations can be conducted concurrently.
- Scalability for phase 2 to thousands of receivers and baselines of up to 3000km, with the possibility of adding new receptor types and possibly new telescopes. Expectation of highly automated operation.

Logically, TM consists of four subsystems:

1. Observation Management, with responsibility for the proposal and observation life cycle.

2. Telescope Management, with responsibility for the instrument life cycle, orchestrating observations, and implementing domain-specific control.
3. Local monitoring and control of TM itself, including responsibility for managing the TM life cycle.
4. Infrastructure that provides the platform for TM, including hardware, power and physical environment. Responsible for dependability aspects: capacity & performance, failover for availability, security etc.

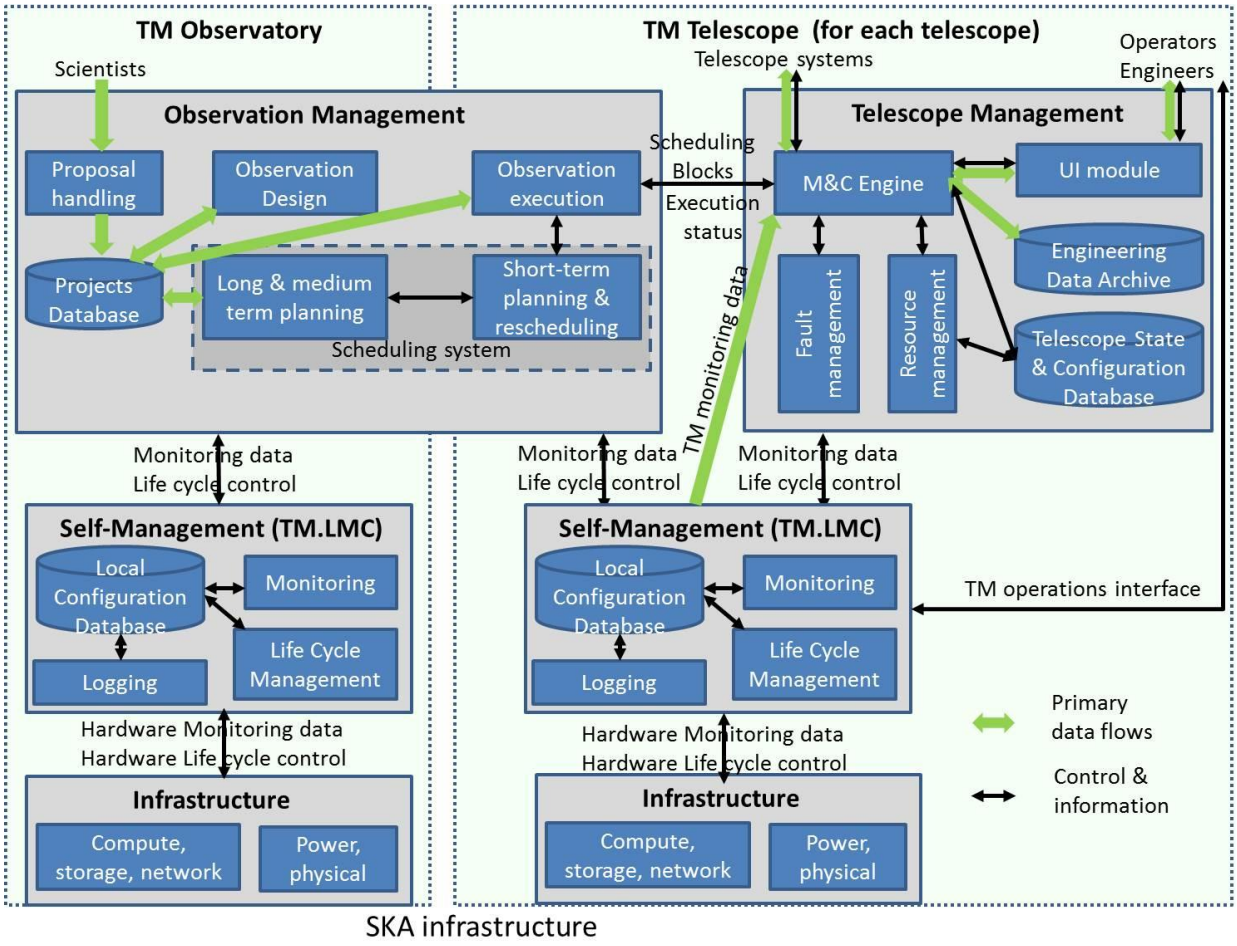

Figure 2. Conceptual view of the overall TM architecture, with some key components and flows

From a PBS (product breakdown structure) perspective, for the purposes of procurement, integration and deployment, TM will consist of three Products. The TM Observatory product will be located at the Global Headquarters and support proposal handling, observation design and long & medium-term planning. SKA-Mid telescope control and SKA-Low telescope control will be located in South Africa and Australia respectively. Each of the Products will require local monitoring and control and infrastructure, and the observation management subsystem is distributed across the Observatory and Telescope products. The Observatory product will also maintain an archive of its engineering data for diagnostic purposes.

Figure 1 shows a conceptual view of the overall TM architecture, showing some of the key components and flows. Some associated components such as the planning tool, observation design tool, forensic tool and log analyzer have been omitted for simplicity.

4. SOME KEY CONCERNS AND APPROACHES

This section discusses some of the key project and architectural concerns and how they are being addressed.

4.1 Standard Monitoring and Control Responsibilities and Interface

TM has interfaces with each of the other SKA systems: SKA-Mid Dishes and MeerKAT Dishes (SKA-Mid only), Low Frequency Aperture Array (LFAA, SKA-Low only), Central Signal Processor (CSP), Science Data Processor (SDP), Signal and Data Transport, including Synchronization and Timing (SADT, SAT), Infrastructure, including power systems, buildings, environment monitoring etc (INFRA). To improve commonality, the SKA architecture defines that each Product belonging to these systems must have a Local Monitoring and Control (LMC) subsystem that is responsible for

- Coordinating and orchestrating the internal functioning of the Product, and presenting a single interface for operational control and high-level monitoring of the Product to TM.
- Monitoring internal parameters, and supplying the monitoring data to TM for presentation to operators and engineers, as well as archiving for diagnostic purposes.
- Detecting faults and providing first level handling of alarms. Alarms are propagated to TM and operators only if they cannot be fully dealt with locally e.g. if human intervention is needed, or if there is a need to notify other affected Products, or correlate alarms across Products. Faults and alarms that are handled locally are also notified to TM for logging purposes. There is a specific guideline that any situation that requires very short response times (typically under 50-100ms) must be handled locally. This is done specifically to avoid real-time systems complexities in TM.
- Ensuring locally safe operation. Each Product LMC has responsibility for ensuring the safety of its own Product, including both people safety and equipment protection.
- Supporting diagnostics and upgrade of Product hardware and software, including maintenance and local control operational modes.
- Providing any domain-specific visualization and engineering activities support.

Standardization of LMC responsibilities leads naturally to the standardization of the interfaces between TM and these LMCs. TM develops the standard interaction model and protocol in consultation with the LMC teams, and then each LMC team defines the details of the specific interface for its Product, since the specific control state machine, list of commands, list of monitoring data, alarms and handling actions etc will be different for each Product type.

This system architectural pattern provides multiple advantages:

- ✓ TM has a uniform interface to all the Products. This allows TM to treat them generically at the architectural and protocol structure levels, with variations being limited to the detailed control and interaction design. This enables substantial reduction in the complexity of TM. This ability to treat products uniformly is augmented by the definition of a *SKA control model*, which defines a common abstract view of all the Products in terms of a standard set of states, operating modes, health statuses etc. Each LMC maps its actual concrete states and statuses to this abstract view for reporting purposes, enabling the definition of a common command set and associated behaviours.
- ✓ Standardization of responsibilities makes it possible to standardize the approach to LMC development, as discussed in the next subsection.
- ✓ The architecture is resilient to variation and change. SKA already has two kinds of dishes (MeerKAT dishes, which are being integrated into SKA, and SKA1 dishes designed specifically for SKA), and it is likely that over time there may be greater heterogeneity in receptor designs, as new and improved technologies are developed. Similarly, other Products can also be expected to evolve considerably over the anticipated 50-year system lifetime. A uniform interface architecture that is applicable to all the diverse Products is likely to remain valid even given unanticipated evolution of each Product.

- ✓ It reduces control system architectural and design complexity, in turn reducing development and operational costs.

4.2 Standardization of Monitoring-and-Control Technologies

For such complex and long-lived systems, life cycle cost minimization makes it highly desirable to reduce technology and monitoring-and-control concept heterogeneity wherever possible. Standardizing LMC responsibilities leads naturally to the possibility of selecting a common control systems framework that can be used to implement the LMCs for all Products.

A unique aspect of the SKA project is that this standardization has been accomplished largely through bottom-up efforts rather than the usual top-down decision-making. Very early on in the project, the TM LMC team initiated the formation of a Community of Practice for LMC developers. Requirements understanding, identification of possible choices, technology evaluation criteria, evaluation results and design ideas and analyses were shared extensively within this network. The result was that when the time came to select the control systems framework technology (soon after PDR), the various LMC teams from different domains and countries were able to very quickly converge on Tango as the framework of choice, over alternatives such as EPICS, ACS (Alma Common Software) etc.

Subsequent prototyping work has shown that indeed Tango is a very good choice. For areas such as infrastructure and network management where the core LMC solution is likely to be provided by an off-the-shelf product that uses a different technology, we will be creating adapter modules.

This Community of Practice approach has now naturally led on to the possibility of sharing common design solutions and implementations. We have formed cross-domain expert teams to investigate and recommend Tango implementation patterns to address common concerns.

4.3 Subarrays and concurrency support

The large number of distributed receivers in the SKA and the high computational capacity naturally lend themselves to performing multiple observations concurrently. This could include situations where a single project partitions the telescope into multiple subarrays that perform related observations concurrently for the purposes of correlated analysis, as well as situations where it is possible to identify multiple independent observations that can be carried out concurrently while still satisfying the requisite quality parameters for each project. SKA also supports commensal observations, where different projects wish to observe the same part of the sky with mutually compatible parameters, and are able to share part or all of the signal processing pipeline. Finally, the scale of the system also makes it likely that at any given time, engineering activities will be carried out on part of the telescope while the rest of the telescope is available for observation. This includes the possibility of continuous commissioning, where new receivers and capabilities are being added to the system without having to take the entire system offline.

Supporting these features requires various TM functions to support partitioning and allocation of telescope resources to concurrent activities. It should be noted that a subarray consists not only of a collection of receivers, but also all the other resources (e.g. along the signal processing pipeline) needed to produce useful data products and engineering outcomes.

- The planning system should be able to identify potential opportunities for concurrency and commensality, either manually or with automation support.
- Observation design must be done with potential concurrency in mind - required sub-array characteristics must be captured and for commensal observing links must be in place to relate observations to each other.
- The scheduling system must be able to reason about what activities can be performed concurrently given available resources. This issue is discussed further in a companion paper[2].
- Observation execution must be capable of executing multiple,. concurrent, observations on different sub-arrays, and for commensal observing ensure that the correct processing is done for each observation and that projects are linked to the correct science data product.

- Telescope management must track the availability of resources, and provide this information to the short-term scheduling function in order to manage concurrently. It must allocate resources to each concurrent activity (subarray) in consultation with the LMCs involved. Once a particular subarray completes its activities, it must release the resources and make them available for the next activity. Concurrently, our expectation is that decisions about concurrency will be mostly made offline, such that the schedule itself specifies which activities to initiate and when, with operators making some modifications in case of contingencies.
- If any faults or problems occur during the observations, observation execution needs to determine which subarrays are affected, and the strategy for continuing with or aborting the various ongoing observations.

In our architecture, the telescope management system views the instrument as a collection of resources to be managed and orchestrated, while the observation management system embeds usage knowledge of how the instrument can be utilized to perform astronomical observations. Accordingly, in the above schema, telescope management focuses on resource allocation and execution of concurrent activities, while observation management determines and manages the use of subarrays to carry out observations concurrently.

4.4 **Capabilities**

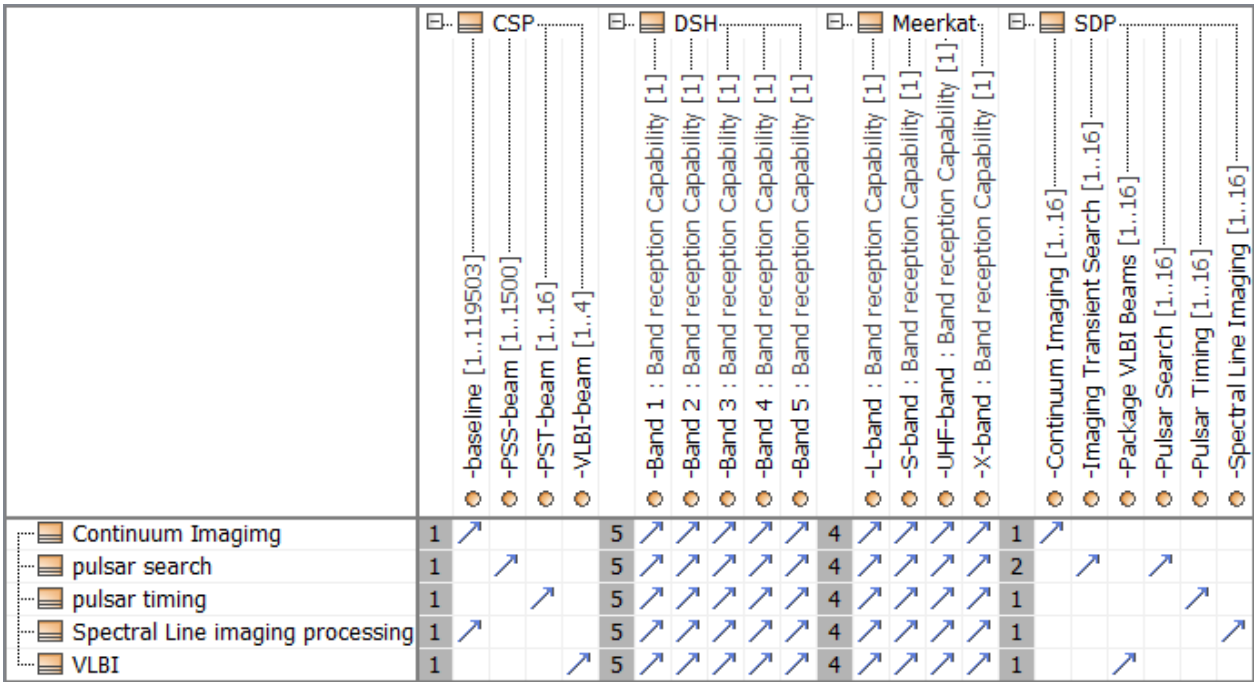

Figure 3. Telescope subsystem ("Element") capabilities required for the various observing modes of SKA-Mid

Observing projects specify desired requirements for observations, such as observing mode, receiver bands, resolution and so on. From these, observation design derives the specifications of the set of resources needed to obtain the data. In order to plan what can be done concurrently, and to identify the specific resources to be allocated to each observation, it is necessary to know the available resources, and enough detailed information about their attributes and behavior to be able to match them up with the demand, including evaluation of all the constraints on concurrency.

For a complex system such as the SKA, this can be an enormously complicated problem. The problem gets even worse when heterogeneity of resources is taken into account. Further, as new technologies arise and each part of the telescope evolves, it is likely that this logic will need to be updated constantly.

To mitigate this problem, SKA has introduced the architectural abstraction of *Capabilities*. The Capabilities of a Product identify the set of functions it can perform to support observations, with parameters that define the associated capacities and variations. Figure 3 shows the collection of Capabilities needed to realize each of the observing modes of the SKA-Mid telescope. For example, continuum imaging operations require band reception capabilities from the MeerKAT and SKA1 dishes corresponding to the desired observing frequencies. The correlator needs to support the required number of baselines (the Capability specification from the correlator identifies the maximum number of baselines that can currently be supported). The SDP system needs to support the processing capabilities needed for continuum imaging. The parameters of each Capability may provide more details e.g. the specific set of transformations supported by the SDP for continuum imaging and associated parameters. Thus the Capabilities model provides a language in which we can identify the specific support needed to carry out a particular observation. Each Product identifies the functions it can support in the same language, so that we can perform matching and allocate a collection of Product Capabilities to observations.

When each Product is commissioned and integrated into the system, it identifies the set of Capabilities supported and their parameters. During operations, it constantly updates its Capability health status e.g. due to internal hardware faults, the correlator may be able to process somewhat fewer baselines - the Capabilities model allows Products to quantify the precise extent of performance degradation in fault situations. For each observation, the Observation Design system identifies the collection of Capabilities required. The planning and scheduling system performs matching between required and available Capabilities to plan the resources needed for each observation, and determine whether and which observations can be performed concurrently. Long and medium term planning is based on the Capabilities expected to be available, while short-term scheduling uses current Capability health status information to check whether a planned observation is feasible given current conditions, and if necessary update the plan to use different resources or select a different observation for execution.

This scheme has two advantages. First, it creates a common language in which all Products can describe themselves, rather than having to hardcode knowledge of each specific Product within the software. With the Capabilities abstraction, we can create planning software that incorporates generic matching logic and takes Product Capabilities as specifications data. As the list of Capabilities evolves over time, relatively few changes will be needed to the software, primarily to accommodate new kinds of functionality and associated characteristics. Second, it simplifies dynamic matching when resources are faulty or degraded. Again, generic matching logic can be created to check schedule validity or perform rescheduling, taking real-time Capability health status information as input.

4.5 Telescope State and Telescope Model

The Telescope Management subsystem collects monitoring data from all the Products and from environment monitoring, and processes this data to derive real-time Telescope State information. This data is made available to any system that requires it, including the operator and engineer UI modules, through a publish-subscribe mechanism. Tango also provides facilities so that in cases where no processing is needed from TM, real-time data can flow directly from the producer device servers to consumers. The engineering data archive subscribes to all data items in order to maintain the historical data logs, suitably decimated.

The real-time Telescope State is one component of the overall system-wide data dictionary, called the Telescope Model. Another component of the data dictionary is various kinds of persistent data, including configuration data associated with each Telescope component, calibration information, and required information such as the Global Sky Model. In addition to data items, the Telescope Model also includes specifications of associated algorithmic logic, such as the pointing model and calibration models, for version management purposes.

ICD mechanisms are used to construct this data dictionary: each Product identifies the information that it produces and consumes, and this is collated to construct the system-wide view.

4.6 Specifications-driven approach

A major design goal in the SKA project is to maximize commonality across the two telescopes in order to minimize life cycle costs. Further, over the long lifetime of the SKA, it can be expected that there will be significant evolution both in system configuration and functionality. and in the underlying realization technological components and

capabilities. The specifications-driven approach is a key architectural pattern to minimize the life cycle costs associated with handling variation and evolution. This pattern involves separating the realization of each component of the software system into two parts: data specifications that capture the logic and behavior of the particular problem at hand, and a generic engine (which can in principle be acquired off-the-shelf) that executes these specifications. For example, monitoring data acquisition can be separated into a generic data acquisition engine, and a collection of specifications that define the set of data items to be acquired, and the acquisition strategy for each item: the source, acquisition method e.g. polling/interrupt, frequency, interaction protocol etc.

The specifications-driven approach allows for independent evolution of the realization engine and the application logic:

- As the telescope configuration evolves, changes in structures, interfaces, and behaviours can be realized by making appropriate changes to the specifications data, instead of having to modify code.
- As realization technology evolves over time, the engine can be replaced with superior versions that interpret the same specifications data.

Nearly every component of TM uses the specifications-driven approach. In many cases, the underlying platforms themselves facilitate this separation e.g. device servers, archivers, loggers, databases, UIs all have toolsets available in which the application content can be supplied as data. The Capabilities concept allows the planning and scheduling to be specifications-driven. With the standard LMC interface, the information about the specific list of commands,, associated state machine, monitoring points, alarms, alarm handling strategies is all populated into a *self-description* data structure (SDD). SDDs are retrieved from each LMC at startup, and used to provide the specifications data to various tools. Orchestration and calibration procedures are written as scripts and plugged in to the system, so that they can be invoked as dictated by the specifications data. In this way, nearly every TM component is designed to be easily evolvable.

The specifications-driven approach also makes it easier to deal with heterogeneity. One of the anticipated challenges in a large system such as SKA is that it is difficult to upgrade all components at once, for various reasons. It can be expected that after a decade or two, there will be considerable variation across, for example, SKA-Mid dishes, both in terms of hardware and software, leading to differences in interfaces, behaviours and the associated control logic. With traditional control software, it can be difficult to deal with so much variation co-existing simultaneously. With the specifications-driven approach, each Product has its own collection of specifications data that control how TM interacts with that Product. TM interprets these different specifications in order to deal with differences in the interfaces and behavior.

4.7 **Self-management**

TM has its own Local Monitoring and Control system that manages the TM life cycle (startup and configuration, execution of control commands from operator, monitoring, fault detection for TM and handling thereof, diagnostics support, upgrades and patches, shutdown), just as the other Products have LMCs. TM LMC also reports monitoring data into Telescope Management, so that it can be archived for diagnostics, and so that TM has a comprehensive and consistent view of the overall telescope status, including itself, that it can provide to operators and engineers.

However, there is a key concern: avoidance of circularity. If Telescope Management can issue control commands to TM.LMC, then there potentially can be situations where if TM is faulty, it will issue bad commands to TM.LMC and prevent it from functioning effectively to detect, report and fix the problem. Further, faults can propagate from other parts of TM to TM.LMC via this control interface. To avoid circularity, there is an architectural restriction that Telescope Management may not issue any commands to TM.LMC. Instead, TM.LMC has a direct interface to operators, on which it reports TM problems, and accepts control commands.

The other major purpose of the self-management system is to increase TM dependability, by detecting any faulty behavior and ensuring that problems get reported to operators and fixed promptly e.g. the offending processes get restarted. For this strategy to be effective, we must minimize common modes of failure between TM.LMC and the rest of TM. TM.LMC needs to run on separate hardware and avoid participating in the direct control of the telescope.

More details about the TM.LMC system can be found in a companion paper[3].

4.8 Scalability

One of the design objectives in SKA is that the solutions we design should be scalable to SKA2, with thousands of receivers and much longer baselines. There are several ways in which we have architected TM to be scalable:

- Hierarchical control: The M&C Engine is designed hierarchically so that there is an aggregation layer between the lowest-level (leaf) nodes within the TM that interact with each Product LMC, and the higher-level subarray controllers within TM that orchestrate the operation of each subarray. As the number of leaf nodes increases as the SKA expands with more dishes and stations, we can aggregate the traffic to/from them to keep the computational load at each level manageable.
- Alarm handling automation, filtering and avoidance of floods: A key challenge in large-scale systems is that we cannot expect operators to deal with every problem manually. With thousands of complex Products to monitor and hundreds of thousands of components, faults and failures will be frequent enough that we must rely on automation to deal with them, and only the most critical problems should be reported for human intervention.

  TM includes facilities for automated alarms handling, alarm filtering and alarms correlation to prevent alarm floods. Operators can set notification preferences to monitor at appropriate levels. Alarms can be configured to automatically raise tickets where needed. Finally, there is a built-in presumption throughout the architecture that some fraction of system elements will be faulty at any given time, and that fault management is an integral part of normal operations.
- Engineering Data Archive: The engineering data archive is required to archive all the monitoring data produced by all Products, which can be highly challenging as scale increases. Scalability has been a core consideration in both the technology choice for the Archive (probably Cassandra) and in the architecture of the archive: each archive node can handle data from a defined group of Products, with queries constructing the required integrated view.
- Capabilities concept: The Capabilities abstraction makes it much easier to handle very large volumes of resources and high heterogeneity, by reducing them to a common language of functions provided, with parameters reflecting the variations. As system scale increases, more time and computational capacity is needed to perform the matching, but there is no increase in software complexity.
- Engineering concurrent with operations: The architecture is designed to cope with continuous commissioning and other engineering activities (diagnostics, maintenance, upgrades) proceeding concurrently with production observations. Discovery mechanisms are built into TM, so that if an entity which has been in engineering mode comes online for production, it will be discovered and integrated, available for future activities. Similarly, faulty entities are isolated and moved into engineering mode, unavailable for production use. Upgrades can be handled by loading and using new versions of self-descriptions. This avoids the need for systemwide downtime to integrate new entities or perform upgrades.

5. CONCLUSION

This paper has discussed some of the key current architectural ideas and concepts for the SKA TM system. It should be emphasized that detailed design is still ongoing, and some evolution and refinement of this architecture should be expected. There are also system level changes and decisions expected that will trigger architecture changes. Accordingly, this paper should be viewed more as a sharing of experiences and current thinking than to serve as a definitive reference for TM architectural concepts.

6. ACKNOWLEDGEMENTS

The TM Consortium is a collaboration between several institutes (NCRA - India, SKA-SA South Africa, STFC - UK, INAF - Italy, CSIRO - Australia, NRC - Canada, IT - Portugal) led by NCRA India and funded from national funding agencies.

REFERENCES

[1] Tango Controls**,** http://www.tango-controls.org

[2] A. Bridger et. al., “Observation management challenges of the Square Kilometre Array", in Proc. SPIE "Software and Cyberinfrastructure for Astronomy IV" (this conference), 9913-35, 2016.

[3] Di Carlo M., Dolci M., Smareglia R., Canzari M., Riggi S. Monitoring and Controlling the SKA Telescope Manager: a peculiar LMC system in the framework of the SKA LMCs, Proc. of the SPIE Astronomical Telescope and Instrumentation 2016, paper no. 9913-117 (this conference)